Stratégies d'encerclement avec information

Nicolas Nisse, David Soguet

Stratégies d'encerclement avec information *

Nicolas NISSE † and David SOGUET

LRI, Université Paris-Sud, Orsay, France

Dans le cadre de l'algorithmique réparti dans les réseaux, l'efficacité d'un algorithme dépend très fortement de la connaissance du réseau, disponible *a priori*. Très souvent, cette connaissance *a priori* est de nature qualitative (taille du réseau, son diamètre, etc.). Fraigniaud *et al.* (2006) ont introduit une mesure quantitative de la complexité d'une tâche répartie dans un réseau. Etant donné un problème réparti, cette mesure, *la taille d'oracle*, consiste en le plus petit nombre de bits d'information dont doit disposer l'algorithme pour résoudre le problème efficacement. Nous nous intéressons à la taille d'oracle permettant de résoudre efficacement *l'encerclement* dans les graphes.

L'encerclement dans les réseaux vise à réaliser la capture d'un fugitif invisible, arbitrairement rapide et omniscient, par une équipe d'agents mobiles, dans un réseau. La stratégie d'encerclement est calculée en temps réel, par les agents eux mêmes, et doit vérifier les trois propriétés suivantes: (1) *connexité* : la zone nettoyée doit toujours être connexe, (2) *monotonie* : la zone nettoyée ne doit jamais être recontaminée, et (3) *optimalité* : le nombre d'agents utilisés doit être le plus petit possible. Les deux premières contraintes assurent des communications sécurisées entre les agents, ainsi qu'un temps de nettoyage polynomial en la taille du réseau. La troisième propriété assure une taille minimum des ressources utilisées. La seule connaissance, concernant le réseau, dont les agents disposent *a priori*, est modélisée par un *oracle* qui répartit sur les nœuds du réseau une chaîne de bits d'information.

Nous prouvons que la taille d'oracle pour résoudre l'encerclement est $O(n \log n)$ bits, avec n la taille du réseau, et que cette borne est optimale. Plus précisément, nous proposons un étiquetage des sommets, de taille $O(n \log n)$ bits, et un protocole réparti utilisant cet étiquetage. Ce protocole permet à une équipe d'agents, dont la mémoire est de taille $O(\log n)$ bits, de nettoyer le réseau de façon optimale monotone et connexe. Ce protocole améliore le protocole proposé par Blin *et al.* (2006) qui ne dispose d'aucune information *a priori* et, de ce fait, nécessite un temps de nettoyage exponentiel. De plus, nous prouvons qu'il n'existe pas de protocole réparti utilisant un oracle de taille $o(n \log n)$ bits qui permette de nettoyer tous les réseaux de façon optimale monotone et connexe.

Keywords: Stratégie d'encerclement, monotonie, bits de conseil

1 Stratégies d'encerclement dans les graphes

L'*encerclement* dans les réseaux (*graph searching*) a été introduit par Parson [10]. Etant donné un graphe dans lequel circule un fugitif invisible, arbitrairement rapide et omniscient, le but est de déterminer une stratégie qui permet à une équipe d'agents mobiles, de capturer le fugitif. Dit autrement, l'équipe d'agents doit "nettoyer" un réseau contaminé. Dans le but de minimiser les ressources nécessaires à la capture, cette stratégie doit impliquer le moins d'agents possible. Les stratégies d'encerclement ont été intensivement étudiées pour leurs applications pratiques (nettoyage de réseaux de pipelines contaminés par un gaz toxique [10], sécurité dans les réseaux informatiques, etc.), mais également pour leurs relations avec des paramètres importants des graphes tels que la largeur arborescente (*treewidth*) et linéaire (*pathwidth*) [2].

Dans le cadre de l'encerclement dans les graphes, un graphe représente un réseau "contaminé" qu'une équipe d'*agents* mobiles doit nettoyer. Initialement, tout le graphe est *contaminé*, et un sommet particulier est désigné comme étant la *base*. Les agents sont situés sur les sommets du graphe et peuvent se déplacer le long des arêtes. Lorsqu'un agent traverse une arête, il la *nettoie*. Une arête *e propre* est préservée de la

†Nicolas Nisse a reçu le support du projet FRAGILE de l'ACI Sécurité Informatique, et du projet GRAND LARGE de l'INRIA.

* Une version étendue de cette article va paraître dans les actes de SIROCCO 2007.

recontamination si, pour tout chemin entre e et une arête contaminée, un agent occupe un sommet de ce chemin. Les mouvements des agents sont contraints de la façon suivante. Initialement, tous les agents se trouvent sur la base, et un agent ne peut bouger que si aucune arête n'est recontaminée à la suite de ce mouvement. Une séquence de tels mouvements, appelés *étapes*, qui permet de nettoyer complètement le graphe est appelée une *stratégie d'encerclement*, ou plus simplement *stratégie*. Le problème de l'encerclement dans les graphes consiste à déterminer une stratégie qui utilise le moins d'agents possible. Une telle stratégie est dite *optimale*. Dans un contexte réparti, que nous considérons dans cet article, le *problème d'encerclement* consiste, étant donnés un graphe G connexe et une base $v_0 \in V(G)$, à déterminer une stratégie optimale de nettoyage pour G de telle sorte que la stratégie est calculée en temps réel par les agents. Notons que dans un contexte centralisé, le problème de l'encerclement dans les graphes est NP-complet [9].

Les contraintes imposées sur les mouvements des agents impliquent deux propriétés importantes des stratégies, i.e. la partie nettoyée ne peut que s'étendre (propriété de *monotonie*), et elle est connexe (propriété de *connexité*). Les stratégies monotones connexes ont été fréquemment utilisées dans le cadre de la conception de protocoles répartis pour nettoyer des graphes [1, 4, 5, 6]. En effet, leurs propriétés assurent un nettoyage du graphe en un nombre d'étapes polynomial en la taille du graphe, et des communications sûres entre les agents. La principale différence entre ces protocoles répartis est la quantité d'information sur le réseau dont les agents disposent *a priori*. Les protocoles proposés dans [1, 5, 6] supposent que les agents disposent d'une information complète sur la topologie du réseau. Ainsi, les agents savent à l'avance dans quel type de réseau ils se trouvent (arbres [1], tores [5], hypercubes [6]). A l'inverse, le protocole proposé par Blin *et al.* [4] permet de nettoyer tous réseaux, sans que les agents ne disposent *a priori* d'information sur le réseau. Cependant, ce protocole ne résout pas complètement le problème de l'encerclement puisque la stratégie qui est effectivement réalisée, est optimale et connexe, mais pas monotone. Ceci est un inconvénient majeur puisque le nettoyage du réseau peut alors demander un nombre d'étapes exponentiel en la taille du réseau. Tous les protocoles précédemment énoncés impliquent des agents avec une mémoire de taille logarithmique en la taille du réseau. Blin *et al.* [4] supposent, de plus, que les nœuds disposent d'une zone locale de mémoire de taille polynomiale en la taille du réseau, accessible par les agents.

Sans surprise, il existe donc un compromis entre la quantité d'information dont les agents disposent *a priori* et la performance de la stratégie réalisée. Nous étudions la quantité minimum d'information dont les agents doivent disposer *a priori* afin de résoudre le problème de l'encerclement dans les graphes. Pour cela, nous utilisons la *taille d'oracle*, mesure introduite par Fraigniaud *et al.* [7], qui permet d'étudier quantitativement l'information qui doit être donnée *a priori* pour résoudre un problème réparti efficacement. Informellement, étant donnés un problème réparti et une instance de ce problème, un oracle fournit une chaîne de bits d'information dépendant de l'instance. Un protocole qui doit résoudre le problème peut se servir de cette chaîne comme d'une information sur l'instance, dont il dispose *a priori*. La taille d'oracle est la longueur maximum de la chaîne fournie en fonction de la taille de l'instance. La notion de taille d'oracle a permis à Fraigniaud *et al.* [7], de différencier quantitativement la difficulté des deux problèmes suivant. Résoudre le problème du réveil dans un réseau de taille n, nécessite un oracle de taille $\Theta(n \log n)$, alors que celui de la diffusion ne requiert qu'un oracle de taille $O(n)$ [7].

Nous prouvons que le problème de l'encerclement dans les graphes de n sommets peut être résolu en utilisant un oracle de taille $O(n \log n)$. Pour cela, nous exhibons un oracle de taille $O(n \log n)$ et un protocole réparti l'utilisant qui résout le problème de l'encerclement en temps $O(n^3)$ dans tous les graphes de n sommets. Nous prouvons que cette borne est optimale : pour tout oracle de taille $o(n \log n)$, il n'existe aucun protocole réparti utilisant cet oracle qui résout ce problème dans tous les graphes de n sommets.

2 Modèle

Les agents sont modélisés par des entités mobiles autonomes, avec des identités distinctes et une mémoire de taille $O(\log n)$ bits où n représente la taille du réseaux. Le réseau est synchrone, et est modélisé par un graphe connexe non orienté. *A priori* le réseau est anonyme, i.e. les nœuds ne sont pas étiquetés. Les $\deg(u)$ arêtes incidentes à un nœud u sont étiquetées de 1 à $\deg(u)$, ainsi un agent peut distinguer les différentes arêtes incidentes à ce nœud. Ces étiquettes sont appelées *numéro de port*. Une instance du problème est un couple (G, v_0), où $G = (V, E)$ est un graphe et $v_0 \in V$ est la base. Un oracle [7] est une fonction O qui

attribue à toute instance (G, v_0) une fonction $f : V \to \{0,1\}^*$ qui assigne une chaîne binaire, appelée *bits d'information*, aux nœuds du réseau. La somme des tailles des bits d'information attribués aux sommets de G est appelée *taille de l'oracle*. Elle permet de mesurer la quantité d'information supplémentaire affectée à G. Le problème d'encerclement consiste à définir un oracle O et un protocole $\mathcal{P}$ utilisant O, avec les caractéristiques suivantes. Pour toute instance (G, v_0), tout sommet $v \in V$ est étiqueté par la chaîne $f(v)$, $f = O(G, v_0)$. Le protocole $\mathcal{P}$, utilisant cet étiquetage, doit résoudre le problème de l'encerclement dans G, avec v_0 pour base. Rappelons que la stratégie est calculée localement par les agents. Ainsi la décision d'un agent sur un sommet v (se déplacer par un certain numéro de port, changer d'état) dépend uniquement de (1) l'état courant de l'agent, (2) l'étiquette $f(v)$ du sommet courant, (3) les états des agents présents sur le sommet (plus si nécessaire le numéro de port d'arrivée si l'agent vient d'arriver sur le sommet). En particulier, les agents ne savent pas à l'avance dans quel graphe ils sont lancés. Les seules informations à propos du graphe sont celles fournies localement à chaque nœud, par l'oracle.

3 Stratégies d'encerclement avec oracle

Nous prouvons tout d'abord le théorème suivant :

Théorème 1 *Le problème d'encerclement peut être résolu en utilisant un oracle de taille $O(n \log n)$ bits.*

Idée de la preuve. Pour prouver ce théorème, nous définissons un oracle de taille $O(n \log n)$, où n est le nombre de sommets du graphe, et un protocole qui résout le problème d'encerclement en utilisant les bits d'information fournit par l'oracle. Le nettoyage du graphe est réalisé en temps $O(n^3)$. Plus précisément, le protocole est constitué de n phases, chacune divisée en deux tours de $O(n^2)$ rondes. Nous décrivons les idées principales de notre protocole. Pour toute instance (G, v_0), nous considérons une stratégie S qui résout le problème de l'encerclement pour le graphe G avec v_0 comme base. La stratégie S induit un ordre $\{v_0, \ldots, v_{n-1}\}$ de nettoyage des sommets, ainsi qu'un ordre de nettoyage des arêtes. L'oracle que nous proposons fournit un étiquetage des sommets qui permet aux agents de distinguer les sommets et certaines arêtes du graphe. Le but de cet étiquetage est de permettre aux agents de nettoyer le graphe en suivant le même ordre de nettoyage des sommets que celui induit par S. Décrivons à présent les arêtes distinguées. Pour tout $1 \leq i \leq n-1$, p_i est l'arête par laquelle un agent, suivant la stratégie S, atteint v_i pour la première fois. d_i est la dernière arête nettoyée de v_i, dans la stratégie S. Soit $T_1 = v_0$ et, pour tout $2 \leq i \leq n$, soit T_i, l'arbre constitué des sommets $\{v_0, \ldots, v_{i-1}\}$ et des arêtes $\{p_1, \ldots, p_{i-1}\}$. Nous prouvons que notre protocole est tel que, au début de la phase $1 \leq i \leq n$, la partie propre G' du graphe G est un sur-graphe de T_i, et un sous-graphe de $G[V(T_i)]$, i.e. le sous-graphe de G induit par les sommets de T_i.

Nous prouvons que, à chaque étape, notre protocole assure que les propriétés suivantes sont satisfaites. Pour chaque sommet propre incident à au moins une arête contaminée, il y a exactement un agent qui *garde* ce sommet, i.e. il protège le sommet de la recontamination. Tout autre agent est dit *libre*. Pour tout $1 \leq i \leq n$, la phase i se déroule de la façon suivante. Les agents libres font un parcours en profondeur d'abord (DFS) de l'arbre T_i, guidés par les étiquettes distribuées par notre oracle. Au cours de ce DFS, les agents libres nettoient certaines arêtes non distinguées de $G[V(T_i)]$ (le choix de ces arêtes n'est pas détaillé ici par manque de place) et, au cours du second tour de la phase i, l'arête p_i pour atteindre v_i. Lors de la première ronde de la phase suivante, une élection a lieu entre les agents pour déterminer lequel d'entre eux devient garde de v_i. Les autres agents restent libres. Un agent qui garde le sommet v_i, $0 \leq i \leq n-1$, ne peut quitter ce sommet que par l'arête d_i. Notre protocole assure également que lorsque ce mouvement est effectué, d_i est la seule arête contaminée du sommet car toutes les autres arêtes incidente à ce sommet ont été nettoyées par les agents libres au cours des phases précédentes. Tous ces mouvements sont effectivement possibles car, l'oracle permet aussi aux agents de savoir à quelle phase les arêtes, distinguées ou non, peuvent être nettoyées (toutes les arêtes non distinguées incidentes à un sommet étant nettoyées au cours de la même phase). Ainsi, l'arête p_i ne peut être traversée qu'à partir du deuxième tour de la phase i. Les arêtes non distinguées incidentes à un sommet sont nettoyées au premier stage de la phase i si la plus grande de ces arêtes est comprise entre p_{i-1} et p_i dans l'ordre des arêtes induit par S. Finalement, un agent qui garde le sommet v_j, $0 \leq j \leq n-1$, le quitte par l'arête d_j au cours de la phase i si d_j est compris entre p_{i-1} et p_i dans l'ordre des arêtes induit par S. Cette dernière opération a lieu au cours du premier ou du second tour de la phase i, selon les cas que nous ne détaillons pas ici, par manque de place.

Nous prouvons que notre protocole résout le problème de l'encerclement en temps $O(n^3)$. L'oracle utilisé est de taille $O(n\log n)$ bits. De plus les agents utilisés peuvent être dans sept états différents, et ont un compteur de taille $O(\log n)$ bits, utilisé pour connaître la phase, tour ou ronde en cours. □

Nous prouvons ensuite que la borne supérieure du théorème précédent est optimale dans le sens suivant:

Théorème 2 *Le problème d'encerclement ne peut être résolu avec un oracle de taille au plus $o(n\log n)$.*

Idée de la preuve. Pour prouver ce théorème, nous exhibons un graphe $\mathcal{G}$, de $O(n)$ sommets. Nous prouvons que toute stratégie pour $\mathcal{G}$ est précisément déterminée. Notamment, nous prouvons qu'il existe $O(n)$ sommets de $\mathcal{G}$ qui doivent être nettoyés dans un ordre précis. Puis, nous considérons l'ensemble $\mathcal{L}$ des *orientations locales* de ce graphe, i.e. l'ensemble des fonctions qui à chaque incidence du graphe (entre un sommet et une arête) attribue un numéro de port. Notons $I = |\mathcal{L}|$. Soit f une chaîne de bits d'information fournie par un oracle. Nous prouvons qu'aucun protocole réparti, utilisant f, ne peut résoudre le problème de l'encerclement dans plus de $I*(\frac{1}{n-2})^n$ instances de $\mathcal{L}$. Ceci est du à la structure très stricte que doit suivre toute stratégie pour $\mathcal{G}$. Soit $\alpha > 0$. Posons $q = \alpha n\log n$ et $t = (q+1)2^q\binom{n+q}{n}$. Nous utilisons un lemme de [7], selon lequel un oracle de taille q bits fournit au plus t chaînes de bits d'information différentes. Pour conclure, il suffit de remarquer que si $\alpha < 1/4$, $I*(\frac{1}{n-2})^n < I*(\frac{1}{t})$ asymptotiquement. Cela signifie qu'il existe une chaîne de bits d'information qui sera attribuée à au moins $I*(\frac{1}{n-2})^n + 1$ instances. Aucun protocole réparti utilisant un oracle de taille $o(n\log n)$ bits, ne peut alors nettoyer toutes ces instances. Notons que ce résultat est indépendant de la capacité mémoire des agents. □

References

[1] L. Barrière, P. Flocchini, P. Fraigniaud, and N. Santoro. Capture of an intruder by mobile agents. In 14th ACM Symp. on Parallel Algorithms and Architectures (SPAA), pages 200-209, 2002.

[2] D. Bienstock. Graph searching, path-width, tree-width and related problems (a survey) DIMACS Ser. in Discrete Mathematics and Theoretical Computer Science, 5, pages 33–49, 1991.

[3] D. Bienstock and P. Seymour. Monotonicity in graph searching. Journal of Algorithms 12, pages 239-245, 1991.

[4] L. Blin, P. Fraigniaud, N. Nisse and S. Vial. Distributing Chasing of Network Intruders. In 13th Colloquium on Structural Information and Communication Complexity (SIROCCO 2006), Springer-Verlag, LNCS 4056, pages 70-84.2006.†

[5] P. Flocchini, F.L. Luccio, and L. Song. Decontamination of chordal rings and tori. Proc. of 8th Workshop on Advances in Parallel and Distributed Computational Models (APDCM), 2006.

[6] P. Flocchini, M. J. Huang, F.L. Luccio. Contiguous search in the hypercube for capturing an intruder. Proc. of 18th IEEE Int. Parallel and Distributed Processing Symp. (IPDPS), 2005.

[7] P. Fraigniaud, D. Ilcinkas and A. Pelc. Oracle Size: a New Measure of Difficulty for Communication Tasks. In 25th Annual ACM Symp. on Principles of Distributed Computing (PODC), pages 179-187, 2006.†

[8] A. LaPaugh. Recontamination does not help to search a graph. Journal of the ACM 40(2), pages 224-245, 1993.

[9] N. Megiddo, S. Hakimi, M. Garey, D. Johnson and C. Papadimitriou. The complexity of searching a graph. Journal of the ACM 35(1), pages 18-44, 1988.

[10] T. Parson. Pursuit-evasion in a graph. Theory and Applications of Graphs, Lecture Notes in Mathematics, Springer-Verlag, pages 426-441, 1976.

† Ces articles ont donné lieu à des versions françaises publiées dans les actes de Algotel 2006.